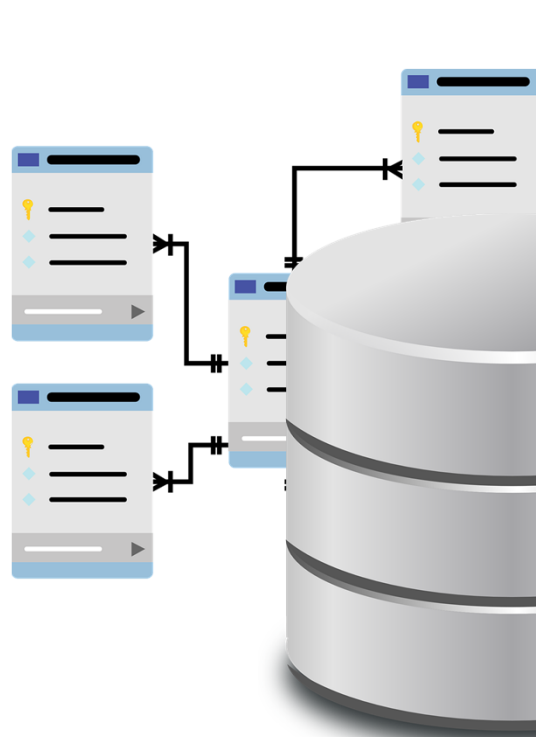


# Introduction to Data Science: CSV Import Using MySQL Workbench



The diagram illustrates the MySQL Workbench interface. On the left, there are three panels: a 'Servers' panel showing a connection to a MySQL server, a 'Query Editor' panel with a SQL query, and a 'Results' panel. The 'Query Editor' panel contains a SQL query that imports data from a CSV file into a table named 'Sales'. The 'Results' panel displays the data imported from the CSV file. The data is presented in a table with 7 columns: Sales\_Date, Day\_of\_Week, Salesman, Temperature, Tweets, Price, and Sales. The table contains 22 rows of data, starting from 1/1/2019 and ending on 1/21/2019.

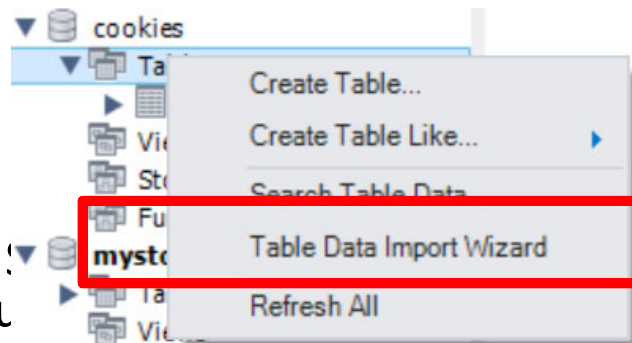
	A	B	C	D	E	F	G
1	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
2	1/1/2019	Tuesday	John	72	2	0.5	177
3	1/2/2019	Wednesday	John	82	3	0.5	127
4	1/3/2019	Thursday	John	69	5	0.5	172
5	1/4/2019	Friday	John	100	7	0.5	150
6	1/5/2019	Saturday	Ada	69	6	0.3	103
7	1/6/2019	Sunday	Ada	91	8	0.5	120
8	1/7/2019	Monday	Ada	81	3	0.3	96
9	1/8/2019	Tuesday	John	88	6	0.5	150
10	1/9/2019	Wednesday	John	69	8	0.5	177
11	1/10/2019	Thursday	John	61	10	0.5	190
12	1/11/2019	Friday	John	79	1	0.5	154
13	1/12/2019	Saturday	John	94	9	0.5	160
14	1/13/2019	Sunday	John	80	5	0.5	161
15	1/14/2019	Monday	John	64	8	0.5	185
16	1/15/2019	Tuesday	Ada	94	6	0.5	137
17	1/16/2019	Wednesday	Ada	68	6	0.3	106
18	1/17/2019	Thursday	Ada	74	2	0.3	85
19	1/18/2019	Friday	John	89	4	0.5	122
20	1/19/2019	Saturday	John	87	6	0.5	187
21	1/20/2019	Sunday	John	100	7	0.5	165
22	1/21/2019	Monday	John	89	8	0.5	158

## Introduction to Data Science: CSV Import Using MySQL Workbench

- Step 1
  - CREATE SCHEMA cookies;
- Step 2

- Step 3

- Convert !  
date valu



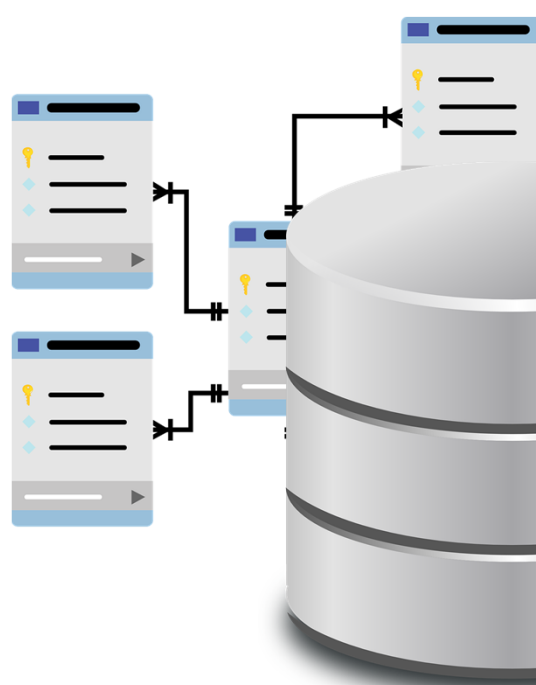
nt to do so or your data include a

**SET SQL\_SAFE\_UPDATES=0;**

**UPDATE** cookies.`cookies sample`

**SET** Sales\_Date = **str\_to\_date**(Sales\_Date, '%m/%d/%Y' );

## SQL Load CSV Data and Covert Date



The diagram illustrates a data loading process. On the left, there are two server icons representing data sources. Arrows point from these servers to a central database cylinder icon. From the database, an arrow points to a table on the right, which represents the target SQL database. The table contains 22 rows of data, with the first row as a header and the subsequent rows as data records. The columns are labeled A through G, corresponding to the data fields.

	A	B	C	D	E	F	G
1	Sales_Date	Day_of_W	Salesman	Temperatu	Tweets	Price	Sales
2	1/1/2019	Tuesday	John	72	2	0.5	177
3	1/2/2019	Wednesd	John	82	3	0.5	127
4	1/3/2019	Thursday	John	69	5	0.5	172
5	1/4/2019	Friday	John	100	7	0.5	150
6	1/5/2019	Saturday	Ada	69	6	0.3	103
7	1/6/2019	Sunday	Ada	91	8	0.5	120
8	1/7/2019	Monday	Ada	81	3	0.3	96
9	1/8/2019	Tuesday	John	88	6	0.5	150
10	1/9/2019	Wednesd	John	69	8	0.5	177
11	1/10/2019	Thursday	John	61	10	0.5	190
12	1/11/2019	Friday	John	79	1	0.5	154
13	1/12/2019	Saturday	John	94	9	0.5	160
14	1/13/2019	Sunday	John	80	5	0.5	161
15	1/14/2019	Monday	John	64	8	0.5	185
16	1/15/2019	Tuesday	Ada	94	6	0.5	137
17	1/16/2019	Wednesd	Ada	68	6	0.3	106
18	1/17/2019	Thursday	Ada	74	2	0.3	85
19	1/18/2019	Friday	John	89	4	0.5	122
20	1/19/2019	Saturday	John	87	6	0.5	187
21	1/20/2019	Sunday	John	100	7	0.5	165
22	1/21/2019	Monday	John	89	8	0.5	158

## SQL Load Data from CSV File

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
▶	1/1/2019	Tuesday	John	72	2	0.5	177
	1/2/2019	Wednesday	John	82	3	0.5	127
	1/3/2019	Thursday	John	69	5	0.5	172
	1/4/2019	Friday	John	100	7	0.5	150
	1/5/2019	Saturday	Ada	69	6	0.3	103
	1/6/2019	Sunday	Ada	91	8	0.5	120
	1/7/2019	Monday	Ada	81	3	0.3	96
	1/8/2019	Tuesday	John	88	6	0.5	150

sales 1 x

**CREATE TABLE** cookies.sales

(Sales\_Date **varchar(10)**,

Day\_of\_Week **varchar(10)**,

Salesman **varchar(10)**,

Temperature **INT**,

Tweets **INT**,

Price **FLOAT**,

Sales **INT**);

## SQL Load Data from CSV File

```
CREATE SCHEMA cookies;
```

```
CREATE TABLE cookies.sales  
(Sales_Date varchar(10),  
Day_of_Week varchar(10),  
Salesman varchar(10),  
Temperature INT,  
Tweets INT,  
Price FLOAT,  
Sales INT);
```

```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/Cookies Sample.csv'  
INTO TABLE cookies.sales  
FIELDS TERMINATED BY ','  
LINES TERMINATED BY '\n'  
IGNORE 1 ROWS;
```

```
(@Sales_Date, Day_of_Week, Salesman, Temperature, Tweets, Price, Sales)  
SET Sales_Date = STR_TO_DATE(@Sales_Date, '%m/%d/%Y');
```

## SQL Load Data from CSV File

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
▶	1/1/2019	Tuesday	John	72	2	0.5	177
	1/2/2019	Wednesday	John	82	3	0.5	127
	1/3/2019	Thursday	John	69	5	0.5	172
	1/4/2019	Friday	John	100	7	0.5	150
	1/5/2019	Saturday	Ada	69	6	0.3	103
	1/6/2019	Sunday	Ada	91	8	0.5	120
	1/7/2019	Monday	Ada	81	3	0.3	96
	1/8/2019	Tuesday	John	88	6	0.5	150

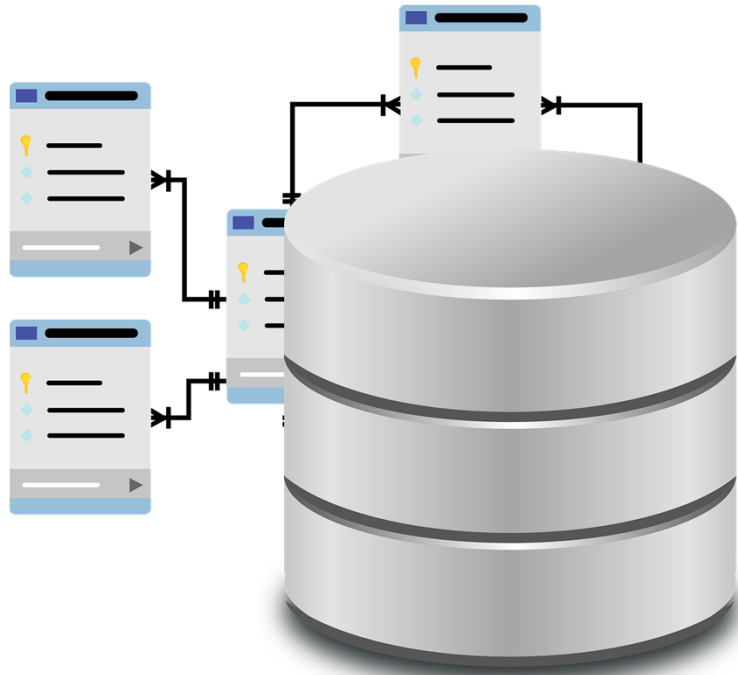
sales 1 ×

Before

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
▶	2019-01-01	Tuesday	John	72	2	0.5	177
	2019-01-02	Wednesday	John	82	3	0.5	127
	2019-01-03	Thursday	John	69	5	0.5	172
	2019-01-04	Friday	John	100	7	0.5	150
	2019-01-05	Saturday	Ada	69	6	0.3	103
	2019-01-06	Sunday	Ada	91	8	0.5	120
	2019-01-07	Monday	Ada	81	3	0.3	96
	2019-01-08	Tuesday	John	88	6	0.5	150

After

## Introduction to Data Science: Data Manipulation Language



## Introduction to Data Science: Data Manipulation Language

`SELECT column from table`

- select data from table

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
►	2019-01-01	Tuesday	John	72	2	0.5	177
	2019-01-02	Wednesday	John	82	3	0.5	127
	2019-01-03	Thursday	John	69	5	0.5	172
	2019-01-04	Friday	John	100	7	0.5	150
	2019-01-05	Saturday	Ada	69	6	0.3	103
	2019-01-06	Sunday	Ada	91	8	0.5	120
	2019-01-07	Monday	Ada	81	3	0.3	96
	2019-01-08	Tuesday	John	88	6	0.5	150



## Introduction to Data Science: Data Manipulation Language

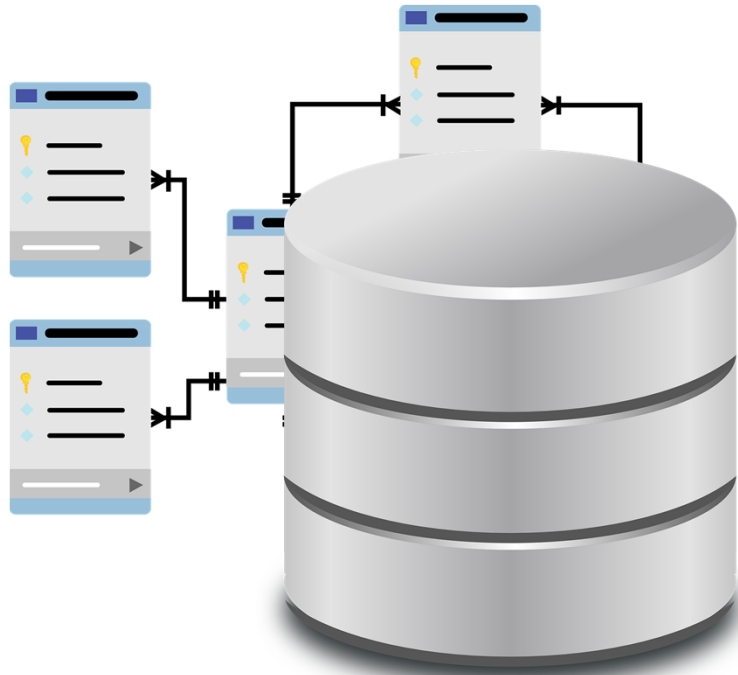
SELECT *column* FROM *table* WHERE *expression*

*Expression* in the form of *column* operator *value*

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
►	2019-01-01	Tuesday	John	72	2	0.5	177
	2019-01-02	Wednesday	John	82	3	0.5	127
	2019-01-03	Thursday	John	69	5	0.5	172
	2019-01-04	Friday	John	100	7	0.5	150
	2019-01-05	Saturday	Ada	69	6	0.3	103
	2019-01-06	Sunday	Ada	91	8	0.5	120
	2019-01-07	Monday	Ada	81	3	0.3	96
	2019-01-08	Tuesday	John	88	6	0.5	150

- select data from table based on the condition specified
- Operator =, <, >, <=, >=

## Introduction to Data Science: DML SELECT AND OR IN LIKE



## Introduction to Data Science: Data Manipulation Language

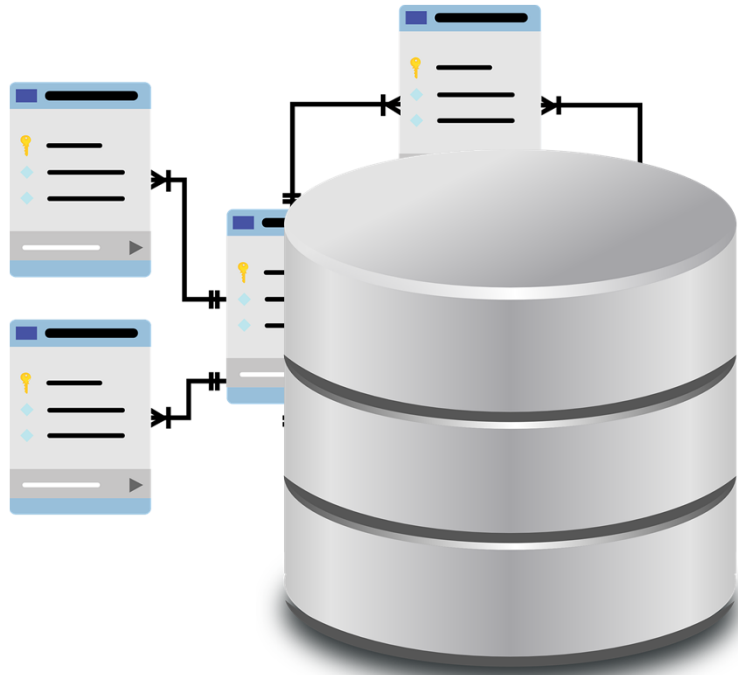
SELECT *column* FROM *table* WHERE *expression*

*Expression* in the form of *column* operator *value*

- select data from table based on the condition specified
- Operator AND, OR, IN, NOT IN, LIKE, NOT LIKE

last_name	first_name	county	district	school	primary_job	fte	salary	certificate
Heckman	William	Atlantic	Atlantic City	Pennsylvania Ave School	Mathematics Grades 5 - 8	1	98774	Standard o
Bird	Kelly	Atlantic	Atlantic City	Atlantic City High School	Coordinator Substance Abuse	1	118415	Standard o
Bean	David B	Atlantic	Atlantic City	Atlantic City High School	Health & Physical Education	0.8	98774	Standard o
Campo	Paula Mia	Atlantic	Atlantic City	Atlantic City High School	Resource Program In-class	1	66184	Standard o
Adams-meyer	Della L	Atlantic	Atlantic City	Atlantic City High School	School Psychologist	1	101866	Standard o
Mansor	Theresa	Atlantic	Atlantic City	Atlantic City High School	Resource Program In-class	1	98774	Standard o
Mendez	Cheryl	Atlantic	Atlantic City	Atlantic City High School	Health & Physical Education	1	109584	Standard o
Toland	John	Atlantic	Atlantic City	Atlantic City High School	Health & Physical Education	1	51696	Standard o

## Introduction to Data Science: Data Manipulation Language



## Introduction to Data Science: Data Manipulation Language

SELECT *column* FROM *table* ORDER BY *column* DESC

- select data from table and sort the result in ascending or descending order

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
▶	2019-01-01	Tuesday	John	72	2	0.5	177
	2019-01-02	Wednesday	John	82	3	0.5	127
	2019-01-03	Thursday	John	69	5	0.5	172
	2019-01-04	Friday	John	100	7	0.5	150
	2019-01-05	Saturday	Ada	69	6	0.3	103
	2019-01-06	Sunday	Ada	91	8	0.5	120
	2019-01-07	Monday	Ada	81	3	0.3	96
	2019-01-08	Tuesday	John	88	6	0.5	150

## Introduction to Data Science: Data Manipulation Language

SELECT *column* FROM *table* WHERE *column*  
BETWEEN *value* AND *value*

- Select data from table based on the interval specified

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
▶	2019-01-01	Tuesday	John	72	2	0.5	177
	2019-01-02	Wednesday	John	82	3	0.5	127
	2019-01-03	Thursday	John	69	5	0.5	172
	2019-01-04	Friday	John	100	7	0.5	150
	2019-01-05	Saturday	Ada	69	6	0.3	103
	2019-01-06	Sunday	Ada	91	8	0.5	120
	2019-01-07	Monday	Ada	81	3	0.3	96
	2019-01-08	Tuesday	John	88	6	0.5	150

## Introduction to Data Science: Data Manipulation Language

**SET SQL\_SAFE\_UPDATES = 0;**

UPDATE *table*

SET *column*=value [, *column*=value,...]

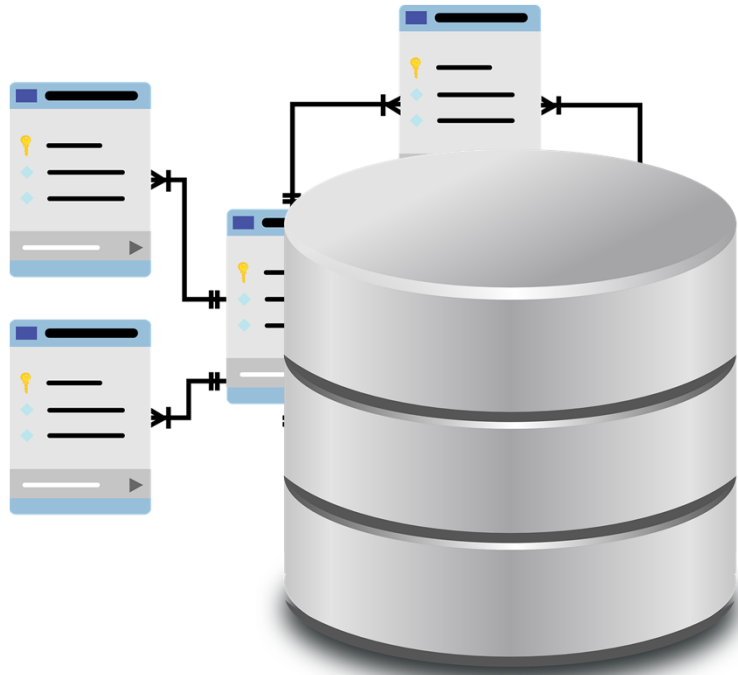
WHERE *expression*

*Expression* in the form of *column* = *value*

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
►	2019-01-01	Tuesday	John	72	2	0.5	177
	2019-01-02	Wednesday	John	82	3	0.5	127
	2019-01-03	Thursday	John	69	5	0.5	172
	2019-01-04	Friday	John	100	7	0.5	150
	2019-01-05	Saturday	Ada	69	6	0.3	103
	2019-01-06	Sunday	Ada	91	8	0.5	120
	2019-01-07	Monday	Ada	81	3	0.3	96
	2019-01-08	Tuesday	John	88	6	0.5	150

- Update table based on the condition specified

## Introduction to Data Science: Data Manipulation Language





## Introduction to Data Science: Data Manipulation Language

SELECT MAX(*column*) FROM *table*

SELECT AVG(*column*) FROM *table*

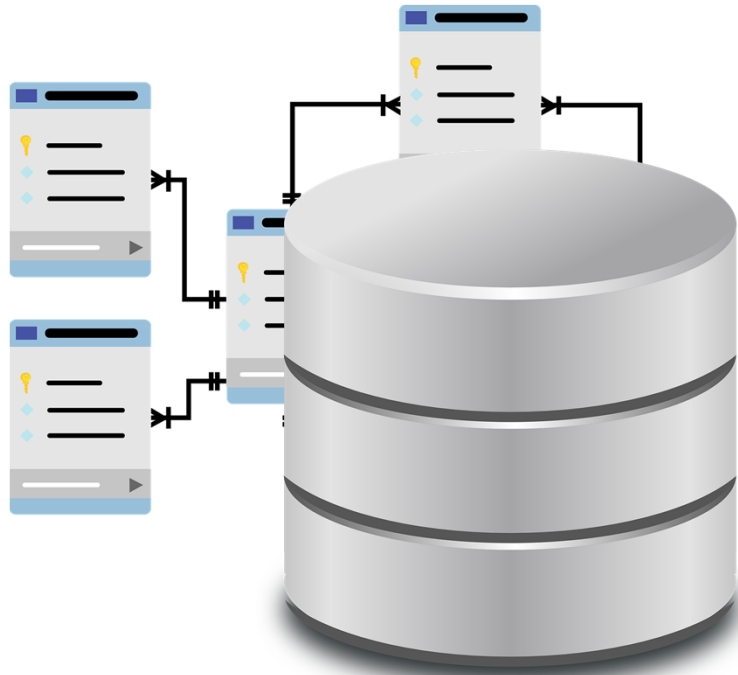
SELECT MIN(*column*) FROM *table*

SELECT COUNT(*column*) FROM *table*

SELECT SUM(*column*) FROM *table*

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
►	2019-01-01	Tuesday	John	72	2	0.5	177
	2019-01-02	Wednesday	John	82	3	0.5	127
	2019-01-03	Thursday	John	69	5	0.5	172
	2019-01-04	Friday	John	100	7	0.5	150
	2019-01-05	Saturday	Ada	69	6	0.3	103
	2019-01-06	Sunday	Ada	91	8	0.5	120
	2019-01-07	Monday	Ada	81	3	0.3	96
	2019-01-08	Tuesday	John	88	6	0.5	150

## Introduction to Data Science: Data Manipulation Language



## Introduction to Data Science: Data Manipulation Language

SELECT *column* FROM *table* ORDER BY *column* DESC

- select data from table and sort the result in ascending or descending order

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
►	2019-01-01	Tuesday	John	72	2	0.5	177
	2019-01-02	Wednesday	John	82	3	0.5	127
	2019-01-03	Thursday	John	69	5	0.5	172
	2019-01-04	Friday	John	100	7	0.5	150
	2019-01-05	Saturday	Ada	69	6	0.3	103
	2019-01-06	Sunday	Ada	91	8	0.5	120
	2019-01-07	Monday	Ada	81	3	0.3	96
	2019-01-08	Tuesday	John	88	6	0.5	150

## Introduction to Data Science: Data Manipulation Language

SELECT *column* FROM *table* WHERE *column*  
BETWEEN *value* AND *value*

- Select data from table based on the interval specified

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
►	2019-01-01	Tuesday	John	72	2	0.5	177
	2019-01-02	Wednesday	John	82	3	0.5	127
	2019-01-03	Thursday	John	69	5	0.5	172
	2019-01-04	Friday	John	100	7	0.5	150
	2019-01-05	Saturday	Ada	69	6	0.3	103
	2019-01-06	Sunday	Ada	91	8	0.5	120
	2019-01-07	Monday	Ada	81	3	0.3	96
	2019-01-08	Tuesday	John	88	6	0.5	150

## Introduction to Data Science: Data Manipulation Language

**SET SQL\_SAFE\_UPDATES = 0;**

UPDATE *table*

SET *column*=value [, *column*=value,...]

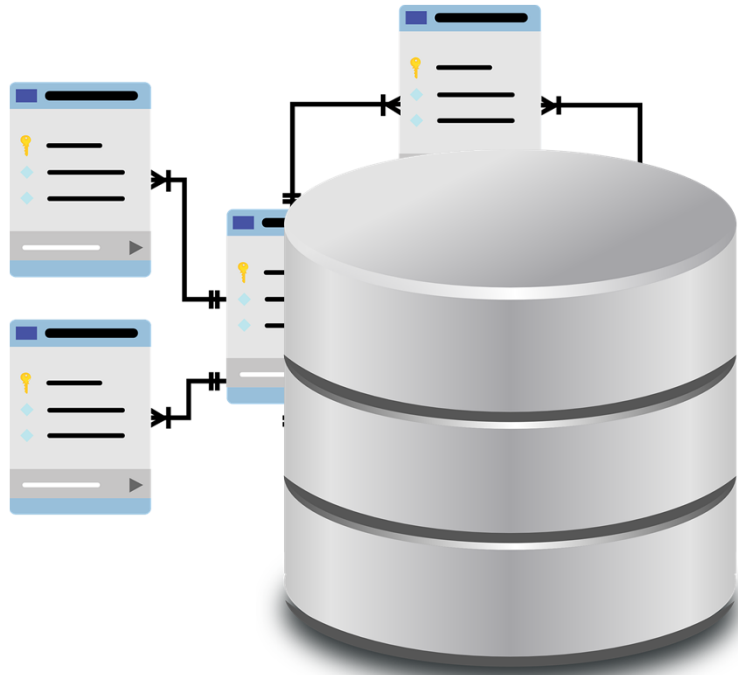
WHERE *expression*

*Expression* in the form of *column* = *value*

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
►	2019-01-01	Tuesday	John	72	2	0.5	177
	2019-01-02	Wednesday	John	82	3	0.5	127
	2019-01-03	Thursday	John	69	5	0.5	172
	2019-01-04	Friday	John	100	7	0.5	150
	2019-01-05	Saturday	Ada	69	6	0.3	103
	2019-01-06	Sunday	Ada	91	8	0.5	120
	2019-01-07	Monday	Ada	81	3	0.3	96
	2019-01-08	Tuesday	John	88	6	0.5	150

- Update table based on the condition specified

## Introduction to Data Science: Data Manipulation Language



## Introduction to Data Science: Data Manipulation Language

SELECT MAX(*column*) FROM *table*

SELECT AVG(*column*) FROM *table*

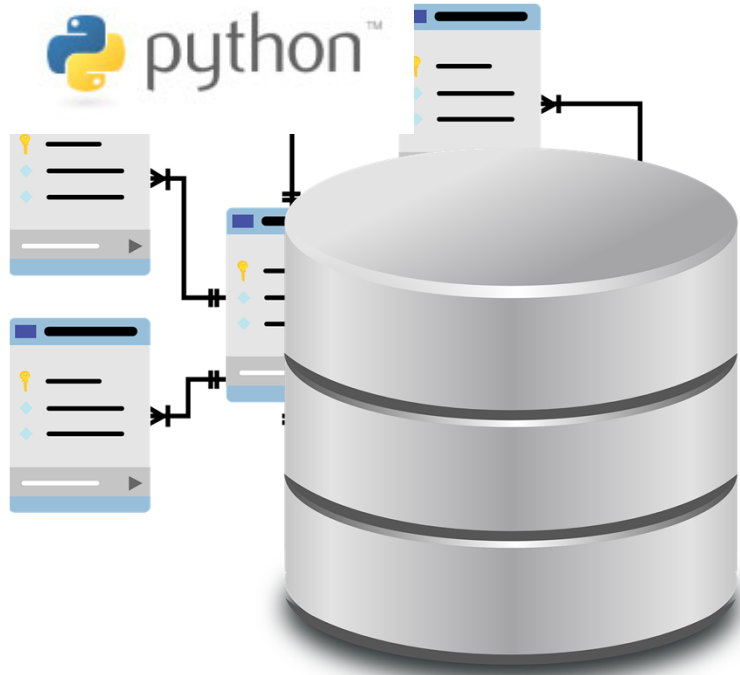
SELECT MIN(*column*) FROM *table*

SELECT COUNT(*column*) FROM *table*

SELECT SUM(*column*) FROM *table*

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
►	2019-01-01	Tuesday	John	72	2	0.5	177
	2019-01-02	Wednesday	John	82	3	0.5	127
	2019-01-03	Thursday	John	69	5	0.5	172
	2019-01-04	Friday	John	100	7	0.5	150
	2019-01-05	Saturday	Ada	69	6	0.3	103
	2019-01-06	Sunday	Ada	91	8	0.5	120
	2019-01-07	Monday	Ada	81	3	0.3	96
	2019-01-08	Tuesday	John	88	6	0.5	150

## Data Science: Python and MySQL Integration Revisit





## Data Science: Python and MySQL Integration

```
('2019-01-16', 'Wednesday', 'Ada', 56, 6, 64.4, 0.3, 106.0, '39.25%')
('2019-01-25', 'Friday', 'Ada', 59, 7, 64.45, 0.3, 84.0, '23.27%')
('2019-01-15', 'Tuesday', 'Ada', 60, 6, 64.4, 0.5, 137.0, '52.99%')
('2019-01-17', 'Thursday', 'Ada', 60, 2, 64.2, 0.3, 85.0, '24.47%')
('2019-01-23', 'Wednesday', 'Ada', 60, 7, 64.45, 0.3, 87.0, '25.92%')
('2019-01-10', 'Thursday', 'John', 61, 10, 85.6, 0.5, 100.0, '14.40%')
('2019-01-14', 'Monday', 'John', 64, 8, 85.5, 0.5, 135.0, '36.67%')
('2019-02-03', 'Sunday', 'John', 65, 6, 85.4, 0.5, 140.0, '39.00%')
('2019-01-06', 'Sunday', 'Ada', 66, 8, 64.5, 0.5, 120.0, '46.25%')
('2019-02-04', 'Monday', 'Ada', 67, 3, 64.25, 0.3, 94.0, '31.65%')
('2019-02-06', 'Wednesday', 'Ada', 68, 5, 64.35, 0.3, 114.0, '43.55%')
('2019-02-11', 'Monday', 'John', 68, 2, 85.2, 0.5, 145.0, '41.24%')
('2019-01-03', 'Thursday', 'John', 69, 5, 85.35, 0.5, 150.0, '43.10%')
('2019-01-05', 'Saturday', 'Ada', 69, 6, 64.4, 0.3, 116.0, '44.48%')
('2019-01-09', 'Wednesday', 'John', 69, 8, 85.5, 0.5, 177.0, '51.69%')
('2019-01-26', 'Saturday', 'John', 69, 0, 85.1, 0.5, 125.0, '31.92%')
('2019-02-02', 'Saturday', 'John', 69, 9, 85.55, 0.5, 128.0, '33.16%')
('2019-01-27', 'Sunday', 'Ada', 70, 6, 64.4, 0.3, 120.0, '46.33%')
('2019-02-05', 'Tuesday', 'John', 70, 4, 85.3, 0.5, 193.0, '55.80%')
('2019-02-07', 'Thursday', 'Ada', 71, 3, 64.25, 0.3, 116.0, '44.61%')
('2019-02-16', 'Saturday', 'John', 71, 3, 85.25, 0.5, 199.0, '57.16%')
('2019-02-19', 'Tuesday', 'John', 71, 2, 85.2, 0.5, 193.0, '55.85%')
('2019-01-01', 'Tuesday', 'John', 72, 2, 85.2, 0.5, 177.0, '51.86%')
('2019-02-10', 'Sunday', 'Ada', 72, 8, 64.5, 0.3, 90.0, '28.33%')
('2019-01-24', 'Thursday', 'John', 75, 1, 85.15, 0.5, 156.0, '45.42%')
('2019-02-15', 'Friday', 'Ada', 75, 4, 64.3, 0.3, 87.0, '26.09%')
('2019-02-13', 'Wednesday', 'John', 78, 0, 85.1, 0.5, 142.0, '40.07%')
('2019-01-11', 'Friday', 'John', 79, 1, 85.15, 0.5, 154.0, '44.71%')
('2019-01-13', 'Sunday', 'John', 80, 5, 85.35, 0.5, 161.0, '46.99%')
('2019-01-30', 'Wednesday', 'Ada', 80, 2, 64.2, 0.3, 105.0, '38.86%')
('2019-02-12', 'Tuesday', 'John', 80, 0, 85.1, 0.5, 174.0, '51.09%')
```

# Data Science: Python and MySQL Integration

```
: import mysql.connector as sq

: mydb=sq.connect(host="localhost",user="root",passwd="ucla", buffered=True)

: mycursor = mydb.cursor()

: mycursor.execute("SHOW DATABASES")

: for db in mycursor:
:     print (db)

: mycursor.execute("select * from supercookies.supercookies")

: print(mycursor)

: result = mycursor.fetchall()

: for row in result:
:     print (row)

: import pandas as pd

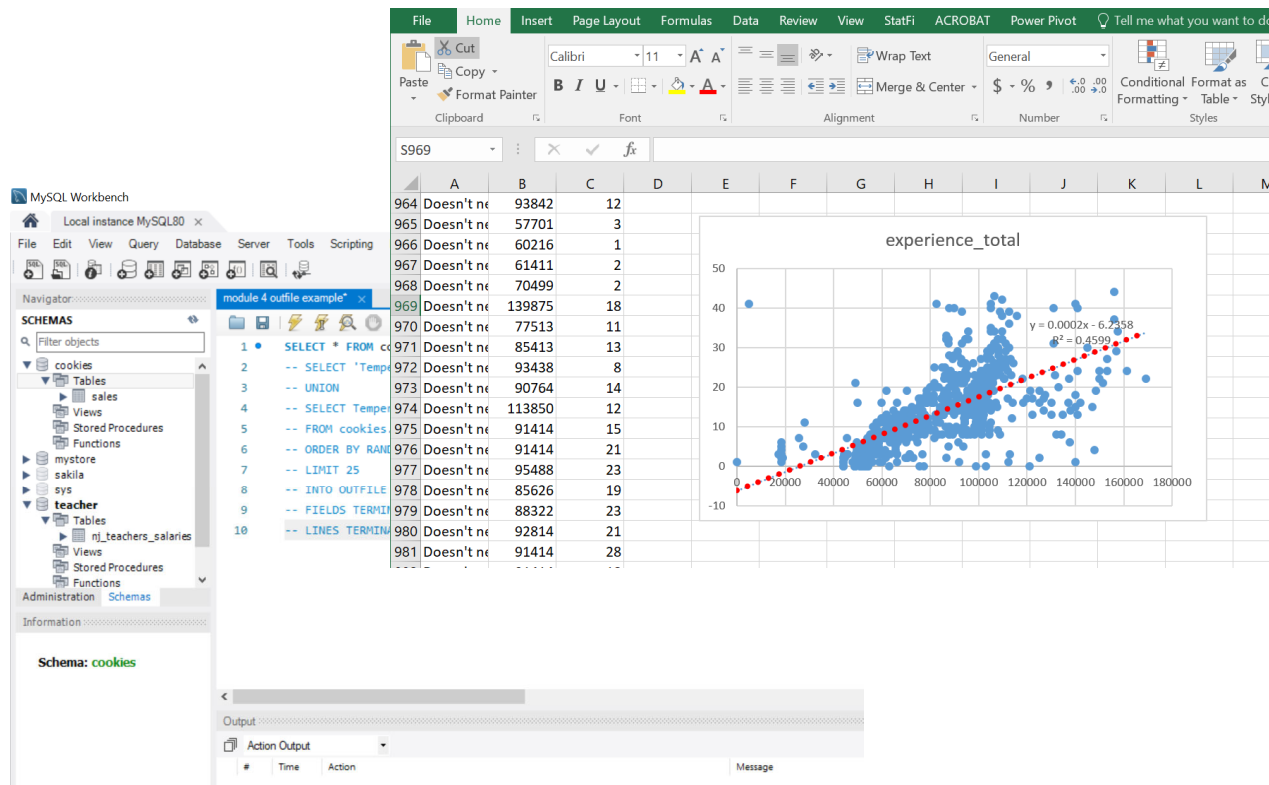
: query = "SELECT * FROM supercookies.supercookies"
: df = pd.read_sql(query,mydb)

: print(df)
```

# Data Science: Python and MySQL Integration

	Sales Date	Day-of-Week	Salesman	Temperature	Tweets	Cost of Good Sold	Price	Sales	Profit
0	2019-01-16	Wednesday	Ada	56	6	64.40	0.3	106.0	39.25%
1	2019-01-25	Friday	Ada	59	7	64.45	0.3	84.0	23.27%
2	2019-01-15	Tuesday	Ada	60	6	64.40	0.5	137.0	52.99%
3	2019-01-17	Thursday	Ada	60	2	64.20	0.3	85.0	24.47%
4	2019-01-23	Wednesday	Ada	60	7	64.45	0.3	87.0	25.92%
5	2019-01-10	Thursday	John	61	10	85.60	0.5	100.0	14.40%
6	2019-01-14	Monday	John	64	8	85.50	0.5	135.0	36.67%
7	2019-02-03	Sunday	John	65	6	85.40	0.5	140.0	39.00%
8	2019-01-06	Sunday	Ada	66	8	64.50	0.5	120.0	46.25%
9	2019-02-04	Monday	Ada	67	3	64.25	0.3	94.0	31.65%
10	2019-02-06	Wednesday	Ada	68	5	64.35	0.3	114.0	43.55%
11	2019-02-11	Monday	John	68	2	85.20	0.5	145.0	41.24%
12	2019-01-03	Thursday	John	69	5	85.35	0.5	150.0	43.10%
13	2019-01-05	Saturday	Ada	69	6	64.40	0.3	116.0	44.48%
14	2019-01-09	Wednesday	John	69	8	85.50	0.5	177.0	51.69%
15	2019-01-26	Saturday	John	69	0	85.10	0.5	125.0	31.92%
16	2019-02-02	Saturday	John	69	9	85.55	0.5	128.0	33.16%
17	2019-01-27	Sunday	Ada	70	6	64.40	0.3	120.0	46.33%
18	2019-02-05	Tuesday	John	70	4	85.30	0.5	193.0	55.80%
19	2019-02-07	Thursday	Ada	71	3	64.25	0.3	116.0	44.61%
20	2019-02-16	Saturday	John	71	3	85.25	0.5	199.0	57.16%
21	2019-02-19	Tuesday	John	71	2	85.20	0.5	193.0	55.85%
22	2019-01-01	Tuesday	John	72	2	85.20	0.5	177.0	51.86%
23	2019-02-10	Sunday	Ada	72	8	64.50	0.3	90.0	28.33%
24	2019-01-24	Thursday	John	75	1	85.15	0.5	156.0	45.42%
25	2019-02-15	Friday	Ada	75	4	64.30	0.3	87.0	26.09%
26	2019-02-13	Wednesday	John	78	0	85.10	0.5	142.0	40.07%
27	2019-01-11	Friday	John	79	1	85.15	0.5	154.0	44.71%
28	2019-01-13	Sunday	John	80	5	85.35	0.5	161.0	46.99%
29	2019-01-30	Wednesday	Ada	80	2	64.20	0.3	105.0	38.86%
30	2019-02-12	Tuesday	John	80	0	85.10	0.5	174.0	51.09%
31	2019-02-18	Monday	John	80	6	85.40	0.5	200.0	57.30%

# Statistics Analysis Using Excel with SQL Randomly Generated Sample



## Statistics Analysis with SQL Randomly Generated Sample

Step 1: Select data from database and save it in a CSV file

Step 2: Use Excel Chart and Built-in statistics functions to analyze data

## Statistics Analysis with SQL Randomly Generated Sample

**SELECT** 'Temperature', 'Sales' **FROM** cookies.sales

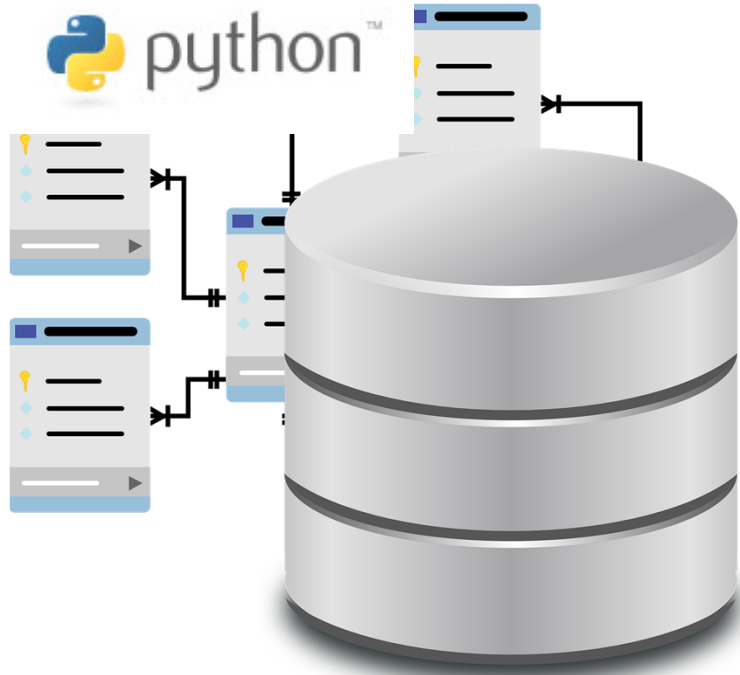
**UNION**

(**SELECT** Temperature, Sales **FROM** cookies.sales  
**ORDER BY** **RAND()**  
**LIMIT** 25)

**INTO OUTFILE** 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/test100.csv'  
**FIELDS TERMINATED BY** ','  
**LINES TERMINATED BY** '\n';

	Sales_Date	Day_of_Week	Salesman	Temperature	Tweets	Price	Sales
▶	2019-01-01	Tuesday	John	72	2	0.5	177
	2019-01-02	Wednesday	John	82	3	0.5	127
	2019-01-03	Thursday	John	69	5	0.5	172
	2019-01-04	Friday	John	100	7	0.5	150
	2019-01-05	Saturday	Ada	69	6	0.3	103
	2019-01-06	Sunday	Ada	91	8	0.5	120
	2019-01-07	Monday	Ada	81	3	0.3	96
	2019-01-08	Tuesday	John	88	6	0.5	150

## Python and MySQL Integration - Revisited



## Python and MySQL Integration

```
import mysql.connector as sq
```

```
mydb=sq.connect(host="localhost",user="root",  
passwd="ucla", buffered=True)
```

```
mycursor = mydb.cursor()
```

```
SQLCMD = 'SHOW DATABASES'
```

```
mycursor.execute(SQLCMD)
```

```
for db in mycursor:
```

```
    print (db)
```

```
('collegeadmit',)  
( 'cookies',)  
( 'covid19',)  
( 'employees',)  
( 'employees_mod',)  
( 'googleapp',)  
( 'govweb',)  
( 'information_schema',)  
( 'mysql',)  
( 'mystore',)  
( 'nj_teachers',)  
( 'nj_teachers2',)  
( 'performance_schema',)  
( 'supercookies',)  
( 'sys',)  
( 'world',)  
( 'youtubeage',)
```



## Python and MySQL Integration

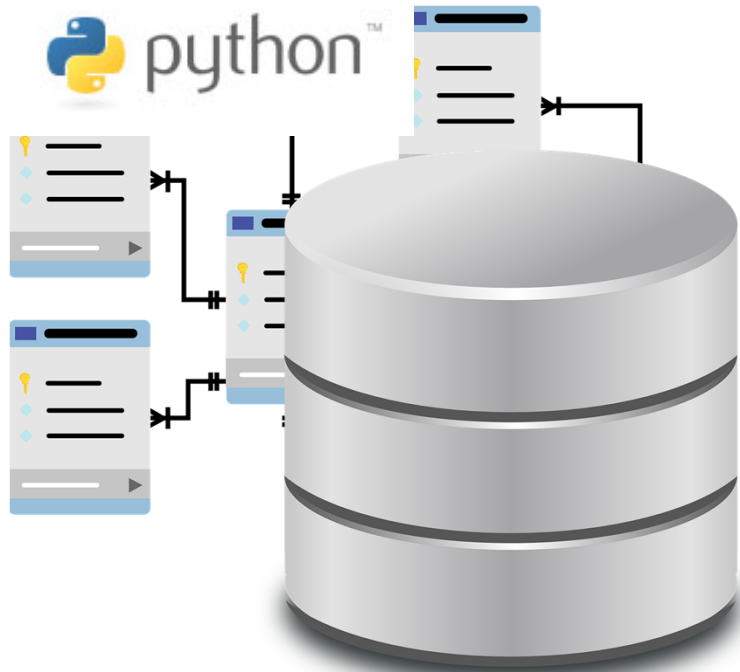
```
import pandas as pd
```

```
SQLCMD = "SHOW DATABASES"
```

```
df = pd.read_sql(SQLCMD, mydb)
```

	Database
0	collegeadmit
1	cookies
2	covid19
3	employees
4	employees_mod
5	googleapp
6	govweb
7	information_schema
8	mysql
9	mystore
10	nj_teachers
11	nj_teachers2
12	performance_schema
13	supercookies
14	sys
15	world
16	youtubeage

## Python and MySQL Integration - Revisited



## Python and MySQL Integration – Data Cleaning

```
import pandas as pd
```

```
import numpy as np
```

```
df = pd.read_csv('cookies dirty data.csv')
```

```
# drop rows if all elements are blank
```

```
df.dropna(how="all", inplace = True)
```

```
# remove leading/trailing spaces
```

```
df["Salesman"] = df["Salesman"].str.strip()
```

```
# replaces invalid values with NaN for one column
```

```
df['Price'] = df['Price'].replace('[^A-Za-z0-9]',np.NaN,regex=True)
```

```
# replaces invalid values with NaN for multiple columns
```

```
df[['Tweets','Sales']] = df[['Tweets','Sales']].replace('[^0-9]',np.NaN,regex=True)
```

```
# drop rows if one or more elements are blank
```

```
df = df.dropna()
```

## Python and MySQL Integration – DateTime Object

	Date	Day	Temperature	Salesman	Tweets	Price	Sales
0	1/1/2019	Tuesday	72.0	John	2	0.5	177
2	1/3/2019	Thursday	69.0	John	5	0.5	172
3	1/4/2019	Friday	100.0	John	7	0.5	150
5	1/6/2019	Sunday	91.0	Ada	8	0.5	120
6	1/7/2019	Monday	81.0	Ada	3	0.3	96

```
df['Date'] = pd.to_datetime(df['Date'], format = '%m/%d/%Y')
```

	Date	Day	Temperature	Salesman	Tweets	Price	Sales
0	2019-01-01	Tuesday	72.0	John	2	0.5	177
2	2019-01-03	Thursday	69.0	John	5	0.5	172
3	2019-01-04	Friday	100.0	John	7	0.5	150
5	2019-01-06	Sunday	91.0	Ada	8	0.5	120
6	2019-01-07	Monday	81.0	Ada	3	0.3	96

## Python and MySQL Integration – DateTime Object

	Date	Day	Temperature	Salesman	Tweets	Price	Sales
0	1/1/2019	Tuesday	72.0	John	2	0.5	177
2	1/3/2019	Thursday	69.0	John	5	0.5	172
3	1/4/2019	Friday	100.0	John	7	0.5	150
5	1/6/2019	Sunday	91.0	Ada	8	0.5	120
6	1/7/2019	Monday	81.0	Ada	3	0.3	96

```
df = pd.read_csv('cookies dirty data.csv', parse_dates=['Date'])
```

	Date	Day	Temperature	Salesman	Tweets	Price	Sales
0	2019-01-01	Tuesday	72.0	John	2	0.5	177
2	2019-01-03	Thursday	69.0	John	5	0.5	172
3	2019-01-04	Friday	100.0	John	7	0.5	150
5	2019-01-06	Sunday	91.0	Ada	8	0.5	120
6	2019-01-07	Monday	81.0	Ada	3	0.3	96

## Python and MySQL Integration – to\_csv

```
df.to_csv("cookies_clean1.csv", index= False)
```